Federal Student Aid (FSA)

eZ-Audit Master Test Plan

Version 1.1

October 21, 2002

Table of Contents

Document Revision History

September 30, 2002 Final document delivered to Randy Wolff, Steve Allison, Janet Scott and FSA Core Team members.

Version No.	Date	Author	Revisions Made
1.0	9/30/02	Matt Williamson	Initial Draft released
1.1	10/07/02	Matt Williamson	SQA comments incorporated
1.1	10/16/02	Matt Williamson	Reviewed SQA comments with FSA

Executive Summary

This document describes how the eZ-Audit application will be tested prior to being migrated into a production environment. Topics covered include unit and assembly testing, system testing, performance (load) testing, and user acceptance testing.

1. The developers will perform Unit and Assembly testing in the development environment.
2. The testers will perform system testing in the test environment.
3. A test environment (database) will be established, and reference/test data loaded into it.
4. The Implementation Team lead will work with the System Test team lead to migrate data into the test database.
5. The System Test team will identify test data requirements (specific reference data that must be established prior to testing).
6. The System Test team will identify scenarios that will need to be tested.
7. The System Test team will manually execute the test scripts.
8. The System Test team and Implementation Team will work together to resolve issues.
9. The software will be functionally accepted, and migrated using Rational Configuration Management standards.

I MASTER TEST PLAN

1 Introduction

1.1 Background

The eZ-Audit implementation is designed to provide a paperless, single-point of receipt and access for financial statements and compliance audits for institutions participating in Federal Student Aid (FSA) Title IV programs. The eZ-Audit application will reduce the cycle-time required to collect and process financial statements and compliance audits from more than 13,500 proprietary, non-profit, and public institutions. The application will enhance the ability of appropriate Ed users to accurately record and report status of annual [audit] submissions. The eZ-Audit application will also improve the quality of FSA service to institutions via the timely acceptance and processing of the audited financial statements and compliance audits.

1.2 Purpose

The purpose of the eZ-Audit Test Plan is to present the test approach for the first release of the eZ-Audit application. The Test Plan defines the business events, test conditions, test cycles, test scripts and scenarios to be tested. It describes the overall testing approach for the eZ-Audit application. The objectives of system testing are to:

- Ensure that a quality product is delivered to the FSA stakeholders
- Minimize risk during implementation
- Find and fix problems during the development process
- Follow FSA guidelines and SLC procedures

This test plan provides an overall framework for testing the eZ-Audit application and is not intended to provide step-by-step instructions (i.e., test scripts) for every aspect of the testing effort.

1.3 Test Scope

The initial release of the eZ-Audit application is intended to allow institutions to submit their annual compliance audit and financial statement submissions electronically. This application will undergo unit, assembly, system, performance (load), and user acceptance testing before being made available to end users. Successful completion of these testing phases will ensure that the new eZ-Audit application meets the business needs of end users as well as the functional and technical requirements specified in the Requirements Deliverable document (refer to Deliverable #86.1.2 eZ-Audit Requirements Document).

1.4 Test Objectives

The objectives of the eZ-Audit Release 1.0 test effort are to:

- Verify the application satisfies the business requirements and supports the eZ-Audit business processes as defined by the eZ-Audit requirements and use cases (refer to Deliverables #86.1.2 eZ-Audit Requirements and #86.1.4 Functional Design)
- Verify the functionality of the application performs as designed.
- Verify the application meets performance requirements.

1.5 Document Objectives

This Master Test Plan for the eZ-Audit Release 1.0 solution supports the following objectives:

- Detail the activities required to prepare for and conduct the testing effort.
- Define the scope of the testing effort, including the items to be tested.
- Document the overall testing approach.
- Define the test environments required to conduct the test.
- Identify and communicate to responsible parties the tasks they are to perform and the schedule to be followed in performing these tasks.

1.6 Test Process

The process used to structure the test activities is illustrated in the diagram below and explained in the following sections. The eZ-Audit System Test team will carry out this process.

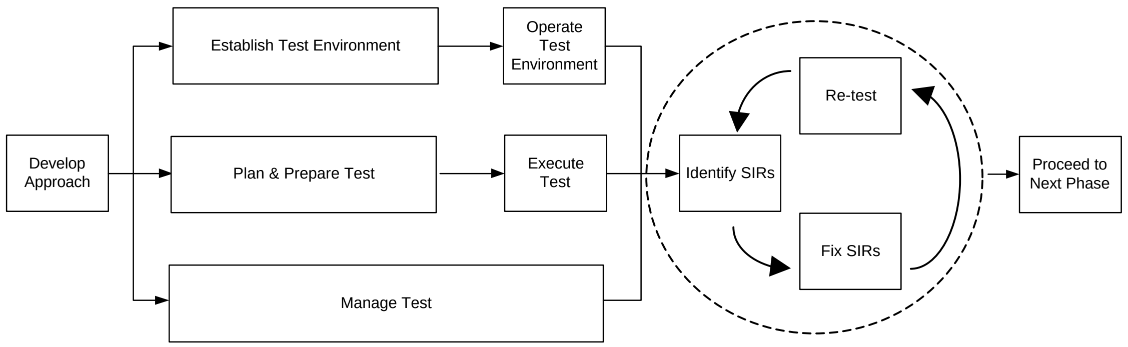


Figure 1, Test Process

1.6.1 Develop Approach

The *Develop Approach* phase consists of the following tasks as defined by the scope of the Master Test Plan:

- Define the test approach and detailed test plan
- Identify the test phases
- Define the scope and objectives of the test phases

- Identify the phase containment strategy and entry and exit criteria
- Identify risks associated with the test effort

1.6.2 *Plan & Prepare Test*

The *Plan & Prepare Test* phase consists of the following tasks:

- Develop the test schedule
- Identify test conditions
- Identify test scenarios
- Define expected results
- Create test scripts
- Define test data
- Verify that the conditions, scenarios, scripts, and data validate functional and technical requirements

1.6.3 *Execute Test*

The *Execute Test* phase consists of the following tasks:

- Execute test scripts for each phase
- Verify expected results
- Log System Investigation Requests (SIRs)
- Update conditions, scenarios, scripts, and data with discrepancies
- Re-test SIR fixes

1.6.4 *Manage Test*

The *Manage Test* phase consists of the following tasks:

- Manage issues
- Facilitate/coordinate test activities
- Track progress against the test schedule
- Report status

1.6.5 *Establish and Operate Test Environment*

The *Establish and Operate Test Environment* phases consist of the following tasks:

- Define the test environment requirements
- Allocate environment resources (both human and physical assets)
- Perform configuration management activities

1.7 Change Control Process

During the solution lifecycle process, the controlled baseline requirements may require updates and the Change Control Process is designed to manage these modifications. Changes may be required for a variety of reasons, including the addition of new technology or functionality, the resolution of issues, and in response to technical and operational tests and evaluations.

The change process for the eZ-Audit Release 1.0 effort is illustrated in the following diagram. This process will be closely followed during the project's Construction (Design, Build, and Test) phase. Identified changes will be logged as System Investigation Requests (SIRs) (see Section

2.2.1 for description of SIRs) and discussed by the Change Control Board (CCB). Changes that are related to existing requirements will be assigned for remediation. Changes that result in a new requirement will be noted as an enhancement and will either be postponed to a future release or will be negotiated with the FSA Project Sponsor for inclusion into the current release.

Please reference the project Configuration Management Plan for additional information on the Change Control Process and the CCB.

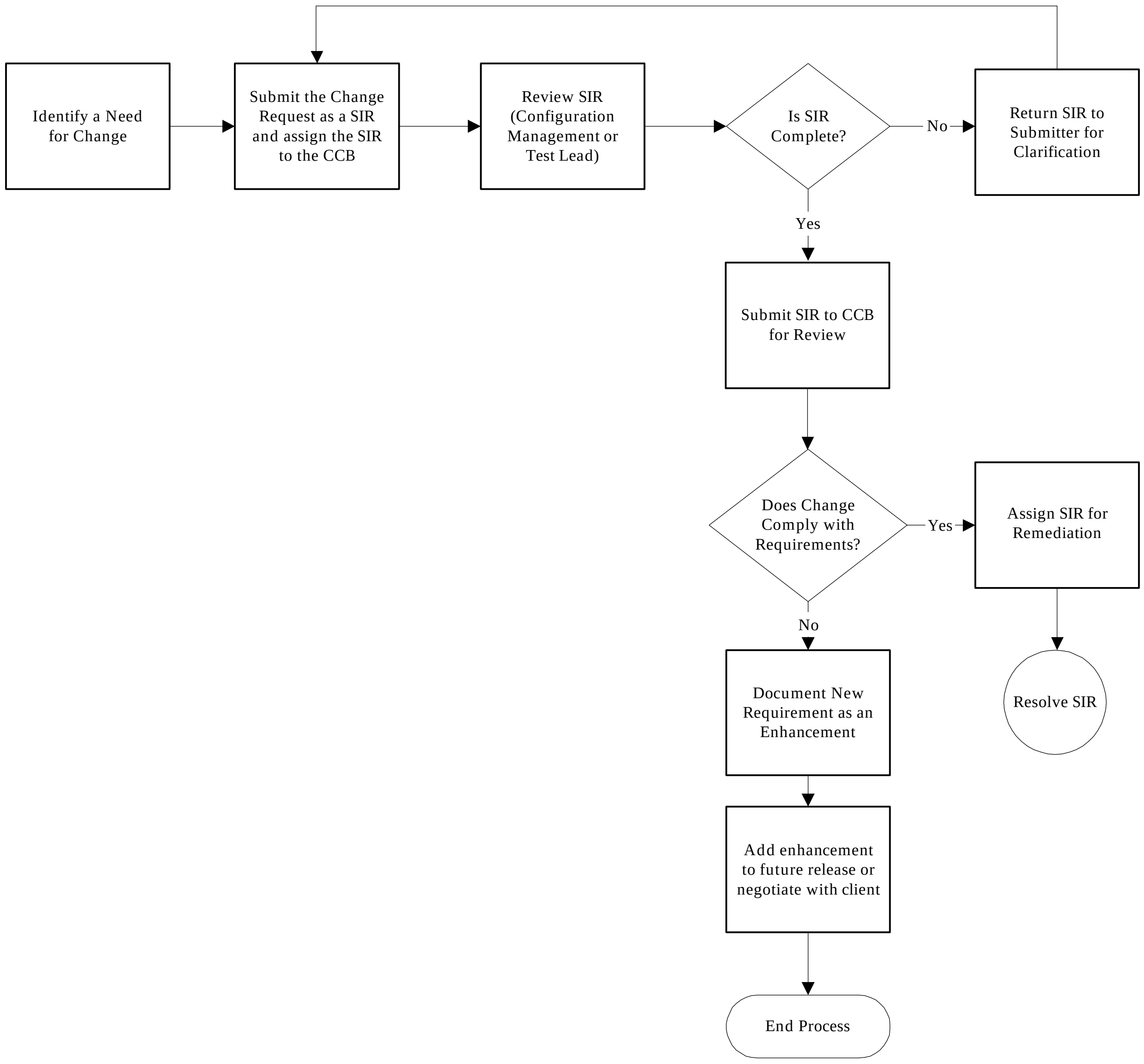

Identify a Need for Change
Submit the Change Request as a SIR and assign the SIR to the CCB
Review SIR (Configuration Management or Test Lead)
Is SIR Complete?
No
Return SIR to Submitter for Clarification
Yes
Submit SIR to CCB for Review
Does Change Comply with Requirements?
Yes
Assign SIR for Remediation
Resolve SIR
No
Document New Requirement as an Enhancement
Add enhancement to future release or negotiate with client
End Process

Figure 2, Change Control Process

1.8 Resources and Responsibilities

The eZ-Audit Implementation team will prepare and execute testing for the unit, assembly, system, and performance test phases.

The eZ-Audit Test team will prepare and execute the test scripts for system testing. The test team will also aid in the facilitation of user acceptance testing, including the preparation and approval of UAT test scripts.

The following user groups will be included in the execution of test scripts for the user acceptance test:

- FSA Case Team members
- Institutions

FSA will assist with the facilitation of the interface system test with the PEPS extract file. In addition, FSA will provide available internal resources and obtain external resources for the user acceptance test.

FSA will perform the Section 508 certification to determine compliance. In addition, FSA will assist in conducting browser-compliant tests with multiple browser versions according to FSA standards.

The following table illustrates the ownership and accountability for each of the test phases.

Test Phase	Ownership / Responsibility		
	Mod Partner		FSA
	DEV	TEST	
Unit Test	X		
Assembly Test	X		
System Test		X	
PEPS interface		X	X
Performance Test	X		X
Section 508 Certification	X		X
Browser-compliance Test	X		X
User Acceptance Test (UAT)		X	X

Table 1, Test Phase Ownership Matrix

2 TEST APPROACH

This section defines the types of testing that will take place (unit, assembly, system, performance and user acceptance testing), and the effort to conduct integration testing with PEPS to retrieve data from the Schools file.

Several phases of testing will be conducted to ensure that the electronic submission process meets the business needs of FSA. The testing will also ensure that the new applications meet the requirements specified in the Requirements Deliverable document. In all phases of testing, the eZ-Audit Implementation and Test Teams will be responsible for identifying test incidents and communicating them to the Implementation Team Lead. These phases are discussed in more detail in the following sections.

2.1 Methodology

There are three key pieces to the testing methodology that will be implemented during the eZ-Audit Release 1.0 test effort. They are the V-Model approach, phase containment, and entry and exit criteria.

2.1.1 The V-Model Approach

The following diagram displays the software development V-Model approach used by the eZ-Audit Modernization Partner team.

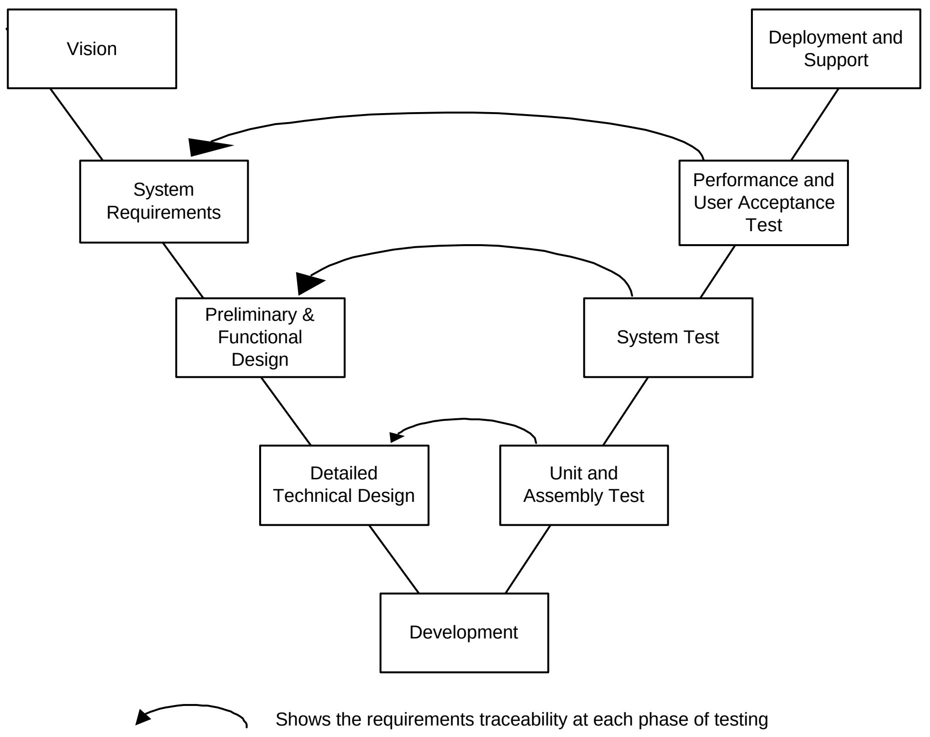

Vision
System Requirements
Preliminary & Functional Design
Detailed Technical Design
Development
Unit and Assembly Test
System Test
Performance and User Acceptance Test
Deployment and Support
Shows the requirements traceability at each phase of testing

Figure 3, eZ-Audit Software Development V-Model Approach

The V-Model provides a structured software development framework, emphasizing building quality into the project starting in the initial requirements phase and continuing through the final testing phase.

The V-Model will serve as the standard approach that will be used to test the eZ-Audit Release 1.0 functionality. The left side of the V-Model contains the requirements definition and design phases; the bottom of the model contains the Development phase, and the right side contains the application test phases.

The requirements and design side of the V-model contains the phases that must be successfully completed prior to development. These phases are: Vision, System Requirements, Preliminary & Functional Design, and Detailed Technical Design. As a software development project moves down the requirements and design side of the model, the application is designed in greater detail with each phase until the design is complete and development begins.

The testing side of the V-Model contains the test phases that a system must successfully complete before it may be deployed to production. The eZ-Audit Release 1.0 test effort will contain five test phases: Unit, Assembly, System, Performance, and User Acceptance Test. According to the V-Model, the level of detail involved in each test phase decreases as the software development project progresses up the right side of the model. As the design phases became more detailed with each passing phase, the test phases begin at the most detailed level and progress to a higher level of detail.

The arrows on the diagram above display the requirements traceability at each phase of testing. The unit and assembly test phases will test the components depicted in the Detailed Technical Design document. The system test phase will focus on the items from the Preliminary & Functional Design documentation, while the performance and user acceptance test phases will test the high-level business processes and scenarios found in the System Requirements documentation.

By implementing the V-Model approach, the following benefits will be obtained:

- Improved quality and reliability
- Reduction in the amount of rework
- Reduction in the cost of problem correction
- Efficient testing by focusing on the objectives of the various tests
- Requirements traceability allowing informed scope decisions
- Improved risk management
- Higher probability of delivery on schedule

In order to achieve these benefits, the processes of phase containment and entry and exit criteria must be implemented along with the V-Model. The phase containment, entry and exit criteria, and the V-Model processes work together to enhance the quality of a deliverable or release.

2.1.2 Phase Containment

Phase containment is used to verify quality deliverables are passed between project phases. The goals of phase containment are to minimize the number and severity of errors passed from one phase into the next and to enable issues to be identified earlier in the testing process, generally resulting in an easier and quicker error resolution process.

To encourage phase containment, progress to the next phase of testing will not occur until a determined exit criteria threshold is achieved.

2.1.3 Entry and Exit Criteria

Phase containment is fostered through the use of entry and exit criteria. For each test phase, entry and exit threshold criteria will be established. Entry criteria are standards used to determine when a software development project is ready to enter a given phase, while exit criteria are standards used to understand what is required of a software development project to exit a particular phase.

The purpose of entry and exit criteria is to improve the quality of the testing effort by acting as an agreement between the Implementation & System Test teams and user community. The criteria will be used to judge software quality levels and readiness for system testing. Entrance criteria will define the conditions that must exist prior to the start of a test execution phase. Exit criteria will define the conditions that must exist prior to the software being released from the testing environment.

The entry and exit threshold criteria for relevant test phases will be reviewed before terminating a test phase and proceeding into the next one. Any deliverable failing to meet its criteria will be returned to its current test phase.

2.1.3.1 Entry Criteria

Entry criteria are standards used to determine when a software development project is ready to enter a given phase. The typical entry criteria for each phase of the eZ-Audit Release 1.0 test effort are:

- Code modules completed – unit and component tested
- Test preparation activities for the particular test phase are complete
- Resources are available to execute tests
- Test environment and test data are in place
- Successful completion of the prior V-Model phase

2.1.3.2 Exit Criteria

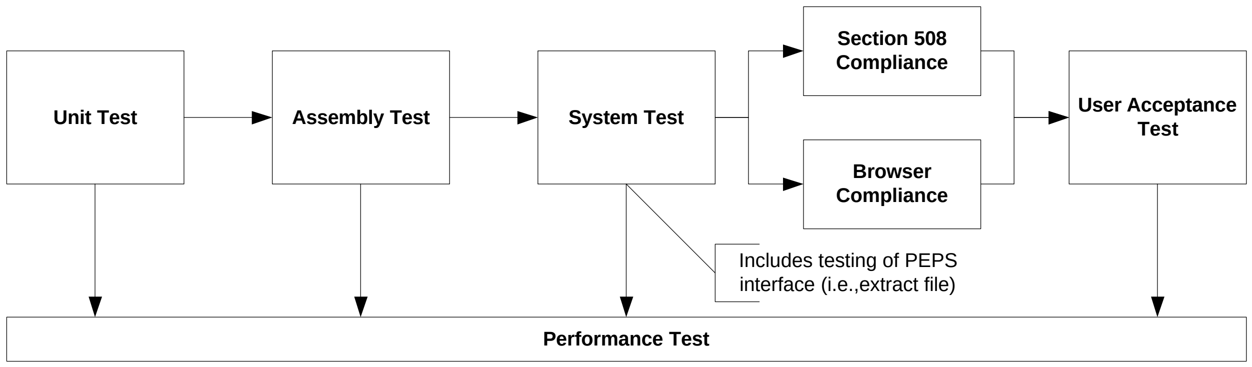
Exit criteria are standards used to understand what is required of a software development project to exit a particular phase. Each test phase must complete the following items:

- Relevant test cases and conditions have been executed
- Identified issues have been properly documented and worked through the resolution process (within the limits specified in Section 2.1 Test Phases).
- All defects have been logged as SIRs in Rational ClearQuest

- Number and Severity of open SIRs are at an acceptable level to move to next phase

If the number and severity of the open SIRs meets the exit criteria, the system will move to the next test phase and the remaining open SIRs will be resolved then. If the product has not met the exit criteria, SIRs will be resolved and tested within the current phase until the exit criteria are met.

2.2 Test Phases



2.2.1 Developer Testing

2.2.1.1 Unit Testing

The eZ-Audit Implementation Team will conduct unit testing to ensure that each developed code module meets its particular business needs and requirements. Unit testing will be conducted in the development environment.

A unit test is the test of an individual component, or module, of the solution. The objective of a unit test is to verify that the component correctly implements the designed specifications.

Unit testing is based on knowledge of how the logical [code] unit is designed to work. The test conditions for the unit test are unique to this effort and will not be repeated in other efforts once the exit criteria have been achieved. Unit testing will include tests for field ranges, values and lengths, functions, data validation, data dependencies, and any special processing contained in the module.

Modules will be developed and become available for testing independently of other modules. The developer responsible for the module will perform the unit test and identify and document the errors related to the independent operation of the program. There will be no SIRs generated by unit testing. Instead, the exit criteria will be the successful completion of all unit test conditions.

2.2.1.2 Assembly Testing

The eZ-Audit Implementation Team will conduct assembly testing, in addition to unit testing, to ensure that the developed code modules meet their particular business need and the requirements. Assembly testing will also be conducted in the development environment.

Assembly test is a string of individual components, or modules, of the application that when combined together test an entire scenario end-to-end. Assembly test scripts are based on knowledge of how the code modules are designed to work together and will mimic the system test functions. The assembly test scripts will be updated if changes occur to the requirements and/or the detailed design specifications.

At the end of assembly test, relevant issues should be identified and documented. Level 1 SIRs (see Section 2.3.1 for description of SIRs) will be fixed and closed prior to progressing to system test. In addition, there should be a minimal number of acceptable open Level 2 SIRs per module – definition of acceptable is at the discretion of the Implementation Manager. No defined limit will be set for Level 3 SIRs while Level 4 SIRs are system enhancements and will not count towards the exit criteria for this phase.

Quantity	SIR Level
0	1 – High
TBD (Dave)	2 - Medium
No defined limit	3 – Low
NA	4 – Enhancement

Table 3, Assembly Test Exit Criteria

2.2.2 System Testing

The system test phase contains three types of testing: System Testing, Accessibility Testing and Browser-compliance Testing.

2.2.2.1 System Testing

System testing concentrates on requirements and business processes, and ensures that the system requirements have been met. System testing will test the components and interfaces working together as a whole, integrated solution.

The eZ-Audit System Test team will handle the creation and execution of test scripts. The test scripts for this phase will be developed using the high-level scenarios developed during the Preliminary and Functional Design phases.

Ideally, system testing cannot begin until all modules have passed both the unit and component test phases because the modules must be fully integrated in the test environment for system testing. However system testing may begin prior to the acceptance of all modules within the prior phases, but the impacted modules must be moved to the end of the test phase. This decision will be at the discretion of the eZ-Audit Implementation & System Test team leads.

The Test Team will conduct system testing on the eZ-Audit application to ensure that all developed code modules work together to meet the intended business needs and requirements. The eZ-Audit System Test team will conduct regression testing to ensure that the components of each system work together as intended. Both regression and system testing will be conducted in the test environment.

2.2.2.1.1 Number of Test Passes

1 First test pass — full test cycle

2 Second test pass — full test cycle, to re-test failed items from first pass and confirm previous successes from first pass after failed items pass successfully

3 Third test pass (if needed) – partial test cycle and regression test, to re-test failed items from the second pass. Anything more than 3 passes probably indicates a serious quality problem, and will be reported to the Project Team Lead. The Test Team will continue to add additional test passes until all test conditions are met successfully.

2.2.2.2 Accessibility Testing – Section 508 Compliance

An accessibility review will be conducted during the system test phase by FSA (within the Office of the Chief Information Officer). The purpose of the review is to verify that persons with disabilities may perform the necessary processes of the eZ-Audit application. All relevant issues should be identified and documented.

Section 508 of the Rehabilitation Act of 1998 mandates this review. Section 508 requires that electronic and information technology developed, procured, maintained, or used by Federal government agencies must be accessible to persons with disabilities. The system must allow a person with a disability to have comparable access to and use of information as a person without a disability.

Section 508 Testing will be conducted on both the front-end screens and reports. FSA must issue a Web Site Accessibility Review with a grade of "PASS" prior to the production deployment of eZ-Audit Release 1.0.

2.2.2.3 Browser-compliance Testing

FSA will test browser-compliance with multiple browser versions according to FSA standards. Browser-compliance testing will test the ability to logon to the eZ-Audit system from multiple browser types and versions. All relevant issues should be identified and documented.

2.2.2.4 System Test Exit Criteria

Ideally, user acceptance testing cannot begin until all modules have passed the system test phase because the modules must be fully integrated in the test environment for user acceptance testing. However user acceptance testing may begin prior to the acceptance of all modules within the prior test phases, but the impacted modules must be moved to the end of the user acceptance test phase. This decision will be at the discretion of the eZ-Audit Implementation & System Test team leads.

At the end of the system test, relevant issues should be identified and documented. Level 1 SIRs (see Section 2.3.1 for description of SIRs) will be identified as In-Progress (refer to SIR Status in section 2.3.3) prior to progressing to the user acceptance test. A determination on the number of open Level 2 and 3 SIRs will be at the discretion of the System Test manager. Level 4 SIRs are system enhancements and will not count towards the exit criteria for this phase.

Quantity	SIR Level
0	1 – High
TBD (Matt)	2 - Medium
No defined limit	3 – Low
NA	4 – Enhancement

Table 4, System Test Exit Criteria

2.2.3 User Acceptance Testing (UAT)

The eZ-Audit Test Team and FSA client personnel will conduct user acceptance testing to ensure that the eZ-Audit application is ready for implementation. User acceptance indicates that the application is ready for deployment to the production-level system, and is the final step in the process of development and testing.

User acceptance testing will be conducted in the test environment. A small number of pre-defined test scripts will be leveraged from System Test to execute UAT. The test scripts to be used for UAT will be approved in advance and documented. The eZ-Audit Test Team will work with FSA to identify UAT participants and organize the testing effort. The UAT participants will actually execute the test scripts while the eZ-Audit Test Team and IV&V facilitates the UAT sessions and is available to answer questions.

Participants will be briefed prior to conducting user acceptance testing; users will be oriented on how to execute the test scripts and log SIRs that they uncover. This effort will provide expected results, repeatable testing conditions and defined exit criteria.

At the end of the user acceptance test, relevant issues should be identified and documented. Level 1 SIRs will be remediated and closed prior to the production deployment of eZ-Audit Release 1.0, while Level 2 and 3 SIRs will be assigned for pre- and post- production remediation by the CCB. Level 4 SIRs are system enhancements and will not count towards the exit criteria for this phase.

Quantity	SIR Level
0	1 – High
TBD (CCB)	2 - Medium
TBD (CCB)	3 – Low
NA	4 – Enhancement

Table 5, User Acceptance Test Exit Criteria

2.2.4 Performance Testing

Performance test is a mechanism for determining an application's performance behavior under load (usage). A goal is usually set for each test, such as the number of users or transactions. Most often, the number of users or transactions is defined based on the maximum usage requirement. The peak usage estimate will be outlined in the Performance Test Plan.

Performance Test also allows for finding a bottleneck in an application; it is another way to fine-tune the application. For applications in development, Performance Test can be used for production capacity planning as well. Performance Testing, at some level, will occur throughout each test phase.

Performance metrics such as the number of estimated concurrent users, response times, transactions per second, acceptable CPU and memory usage, etc. will be defined in a more detailed Performance Test Plan to be delivered in January 2003 based on requirements already captured.

At the end of performance test, relevant issues should be identified and documented. Level 1 SIRs should be remediated and closed prior to the production deployment of eZ-Audit Release 1.0. Open Level 2 SIRs will be assigned for pre- or post- production remediation by the CCB, while Level 3 SIRs will be at the discretion of the Implementation manager. Level 4 SIRs are system enhancements and will not count towards the exit criteria for this phase.

Quantity	SIR Level
0	1 – High
TBD (CCB)	2 - Medium
TBD (Dave)	3 – Low
NA	4 – Enhancement

Table 6, Performance Test Exit Criteria

2.3 System Investigation Requests (SIRs)

Whenever a test condition or test script step does not return the expected result, the tester will log the discrepancy between the expected and actual result as a System Investigation Request, or SIR. SIRs will be logged and tracked in a central repository, or SIR database, using Rational ClearQuest. The SIR database will include a long and short description of the issue (including the actual and expected results), its severity and status, the affected test area or module, the test phase, test pass, and tester.

2.3.1 SIR Severity

The following table contains the SIR severities and definitions. The tester will use these definitions to provide an initial severity, but the eZ-Audit Test Lead and the CCB will have the authority to make a final decision on SIR severity levels.

ID	Label	Description
1	**High**	The anomaly results in a failure of the complete software system, a subsystem, or a software module within the system. There is no way to make the failed component(s) work; however there is an acceptable processing alternative that will achieve the desired results (an acceptable work around exists).
2	**Medium**	The anomaly does not result in a failure, but causes the system to produce incorrect, incomplete, or inconsistent results, or the anomaly impairs system usability.
3	**Low**	The anomaly does not cause a failure, does not impair usability, and the desired processing results are easily obtained by working around the anomaly. The anomaly may also be the result of non-conformance to a standard or related to the aesthetics of the system.
4	**Enhancement**	The anomaly is a request for an enhancement. Anomalies at this level may be deferred to a future release or will be treated as an addition to the defined project scope if the Modernization Partner and FSA agree upon its inclusion.

Table 7, SIR Severity Levels

2.3.2 SIR Status

The following table contains the SIR Status levels that will be used during this effort and a definition of each:

SIR Status	Definition
Opened	The SIR was created and is awaiting assignment or review
In-Progress	The SIR was assigned and that resource is currently working on its resolution
Resolved	Corrective action was completed and the SIR is awaiting validation
Closed	Corrective action has been completed and validated
Postponed	The CCB agreed to postpone the corrective action on this SIR to a later release
Duplicate	The issue identified in this SIR is identical to another open SIR, and the resolution to the duplicate SIR will be tracked through the original open SIR

Table 8, SIR Statuses

2.3.3 Re-test SIR Fixes

As SIR fixes are completed, the script in which the SIR was encountered will be re-tested. These SIRs may have originated from an earlier pass, the same pass, or from another script that has one or more components in common. The System Test Lead will coordinate with the Implementation Lead to group related issues together, thus minimizing the delay of executing the next pass of the test script.

SIR fixes will be re-tested using local copies of the original input files, data, etc. After conducting a re-test, SIRs that fail are reopened and newly identified SIRs are logged in the SIR database. Additional passes will continue to be conducted, as appropriate, until the exit criteria are achieved for each phase.

II DETAILED TEST PLAN

3 Test Plan Structure

The process for developing system test scripts will be as follows:

1. Identify Test Scenarios from Requirements Definition documents (e.g., Preliminary and Functional Design, Use Case Specifications)
2. Write Test Scripts for test execution
3. Identify Test Conditions from Detailed Design documents
4. Map Test Conditions to Use Cases [which trace to Requirements]
5. Create baseline Test Data
6. Conduct Test Readiness Review (TRR) session

3.1 Test Scenarios

Using the Use Case Specifications, test scenarios will be identified to demonstrate how users perform specific tasks in the eZ-Audit application. Use Case specifications describe functionality in terms of user behaviors to be satisfied by the system. Therefore, the test scenarios will describe the user interaction with the system required to perform a specific business function.

3.2 Test Scripts

The test team lead will work with the test team to create test scripts. Reports can then be generated and given to the test team showing the conditions that a script should test, and the tester will write or modify a script to cover those conditions. Scripts should be consolidated, wherever possible, to maximize test scenarios and minimize staff hours.

Test scripts are executed within each test cycle. The scripts test the basic system functionality derived by the system level requirements. Test scripts are documented in terms of a business function or technical requirement and serve as the basis for creating the test conditions.

3.2.1 Test Script Standards

Please see Appendix B for an example of what the eZ-Audit system test scripts will look like. Each test script contains the following information:

1. Script Name & Number – assigned by script writer
2. Description
3. Created By
4. Tested By – person who is performing the actions for a test condition

5. Test Date – written on the printed script by the tester
6. Prerequisites – reference data or transactions that must exist prior to script execution
7. Use Case(s) Covered

For each step within the script, the following information is shown in table format:

8. Step number – auto-assigned by the script template
9. Step Description – describes what the tester needs to do for every step of the condition
10. Action – detailed instructions for how to accomplish the step
11. Input – describes the data that needs to be entered into eZ-Audit at each step
12. Expected Results – describes how the system should respond to the action
13. Actual Results – describes what happens when the tester performs the action
14. Pass/Fail – describes whether the expected and actual results match for a condition. Pass means that the expected and actual results match. Fail indicates the expected and actual results did not pass for a condition.
15. Priority – describes the conditions that fail, the severity of the defect and the priority for development to fix the defect.
16. Comments – notes about the test condition
17. SIR # - related SIR # for a defect logged against the failed test condition. If the condition fails for more than one reason, multiple SIRs will be logged.
18. Requirement # – traces the test condition to a system requirement

3.3 Test Conditions

Test conditions provide the ability to test the system functionality under specific circumstances to be performed in production. The test conditions will be developed so that they are repeatable and reusable. Test conditions must be repeatable so that errors can be reproduced and retested under the same circumstances.

Test conditions will be written using the detailed functional and technical requirements from the Requirements Definition phase. These conditions will be grouped into scenarios. The scenarios will contain a related set of conditions, with the condition mapped to the Use Cases that trace to the detailed requirements.

Finally, test scripts will be prepared during the Application Development phase that define the steps, input data, and expected result to test the scenarios. A single scenario may be split across multiple test scripts, or multiple scenarios may be included within a single test script.

3.4 Requirements Traceability

The test scripts and test conditions will then be mapped to the Use Case Specifications that already trace to the related detailed system requirements in the Rational database.

3.5 Baseline Test Data

Baseline data must be defined and loaded into the system prior to testing. This data includes:

1. eZ-Audit Test Data – audit and financial data
2. User Setup Data – user roles and responsibilities, audit finding codes, etc.

Test data will be created as representative sample data to use when executing the test scripts. Before test execution can begin, baseline test data will be created to enable the testers to execute the test scripts. The 'Input' column will represent data entered into eZ-Audit while executing the test scripts. This 'Input' data needs to be refreshed prior to the start of each Test Pass and the 'Baseline' data needs to be reapplied to each System Test instance. Required fields will be bolded in the test scripts and the respective field names in eZ-Audit will contain asterisks.

Data can be used only once during test execution and will need to be refreshed at the end of each Test Pass. Although, new test data or different baseline data may be used to conduct multiple passes of a single test script within each Test Pass. This allows the tester to do parallel test execution since the data being manipulated has already been created and saved to the database.

It is still necessary to identify the test entry data needed in order to execute the test scripts when the scripts require the tester to create data from scratch. For example, a tester will need to enter financial data to execute one condition and then calculate the composite score to fulfill another condition. This type of iterative testing ensures that the data being entered is valid and allows the tester to complete an entire Test Cycle using the same data entity.

3.6 Test Readiness Review

Test conditions, test scripts, execution timeframes and testing phases will be reviewed during the Test Readiness Review. All parties involved or impacted by the eZ-Audit testing effort will attend the Test Readiness Review. Attendees include FSA Core Team, the eZ-Audit System Test team, FSA stakeholders and IV&V.

4 System Test Execution

4.1 System Test Environment

A separate test instance will be set up for the test team. All successful unit-tested and component-tested code modules will be migrated to the test environment prior to test execution. Two test instances will be created. Having a second test instance will allow the test team to create a new baseline each time a test pass has been completed.

The first test instance will be the current testing instance used to execute each System Test Pass. The second instance will be used solely for inputting baseline test data. At the beginning of each test pass, the System Test Lead needs to coordinate with the Tech Arch Team to refresh both instances and apply the baseline data to each instance.

4.2 Test Execution

The System Test team will conduct testing in a team environment. Experience has shown that having all testers in the same room leads to quicker issue resolution and greater learning than having the testers dispersed. Start date TBD, pending final approval of detailed test plan.

The System Test team will execute the scripts according to the step-by-step directions on each script. Scripts that pass will be signed off and stored in a binder. Scripts that fail should be dealt with as described in the following section.

4.2.1 Test Passes

A test pass is defined as one full test execution cycle. Development remediation activities are conducted between passes within the same test phase. Additional passes will be run within a test phase until the test meets the defined exit criteria.

Test passes will be performed in steps. A pass contains the following steps:

1. The tester executes the test scripts.
2. The tester enters a SIR when results from the test deviate from what is expected.
3. The test lead notifies the development lead when the test pass is complete.
4. The development lead runs the SIR report at the end of the pass.
5. The development lead performs an impact analysis on the SIR list and reports his/her findings to the test lead.
6. SIR fixes are completed and moved into the testing environment according to Configuration Management guidelines.
7. The development team updates the SIR database with the SIR status information.
8. The development lead notifies the test lead that testing may resume.
9. The test team begins a new pass and continues this process until the exit criteria have been achieved for the phase.

If a critical error occurs within a test pass that prevents the entire set of scripts from being executed and the fix requires lengthy remediation, the pass will be considered complete and the

SIR resolution phase will begin. Once the SIR fixes are migrated into the test environment, a new pass will begin. The beginning of a new pass will be at the discretion of the System Test lead, and a new pass may begin with unresolved SIRs outstanding. Test passes will continue until the scripts for that pass have been executed and the exit criteria have been satisfied. The amount of time necessary to complete a test pass will vary depending on the particular area of testing being performed.

4.3 Identification and Resolution of Issues

The following process will be followed for identifying and resolving problems encountered during testing.

1. The Tester discovers a discrepancy with a script and works with another tester to determine if the problem is with the software, the test script, or some other source. This will serve as a form of triage to identify true issues, and to not bother the programmers with issues not related to the code.
2. If the test team determines that the problem is with the software, the tester will log the issue in the SIR database with supporting detail including, screen prints or sections of the requirements document. The tester will document the problem by entering "Fail" in the Pass/Fail column of the script, at the step where the script failed. The tester should also capture the date and time the failure was noted.
3. The SIR database should be updated to reflect the script failure. In addition to tracking software failures, the team will also track failures related to undefined requirements, faulty scripts, Oracle errors, and "other."
4. The test team lead will notify the Implementation Team lead of any failed test scripts in order to assign the problem to a developer to correct it.
5. The developer should work with the tester in the development environment to ensure that the software change fixed the problem.
6. The Implementation Team lead should notify the Test team lead when the changed software has been migrated to the test environment. This should be done following the FSA-defined process.
7. The tester will confirm, in the test environment, that the software change fixed the problem.
8. The tester will note, in the Pass/Fail column of the script, the date and time that the step of the script was successfully executed, and continue with the script.

During this process, if a problem is identified in testing, the System Test and Implementation Team leads will make a determination on the need for and scope of regression testing.

The key to quickly resolving software defects is direct communication between the tester and the developer who is fixing the problem. The Implementation Team lead and the System Test team lead should be kept in the loop, but must not become a bottleneck.

4.4 Monitor and Report on Testing Progress

Testing progress will be tracked by business function, rather than module, because certain functions cannot be tested until additional modules are delivered to the testing team. If we track our progress based on business function, then we can state that every function of a screen has been tested *except* for those related to a particular module. If, on the other hand, we track our progress by module, then the screen as a whole cannot be considered fully tested until all modules that affect it have been delivered and tested.

The test script development database will be used as a tracking tool to keep current on:

1. Total scripts to test
2. Scripts tested to date
3. Tester name
4. Date tested
5. Pass/Fail
6. Problems found, by category (software, database, ill-defined requirements, *etc.*)
7. Solution
8. Date Re-tested

The System Test team lead will provide bi-weekly updates of completed scripts, success / correction percentages to the project lead and Implementation team lead. The System Test team lead will also update the project plan and provide written notice to the Implementation team lead of repeated errors.

5 Development, Test, and Production Environments

The following diagram depicts the environments that will be required for the eZ-Audit project.

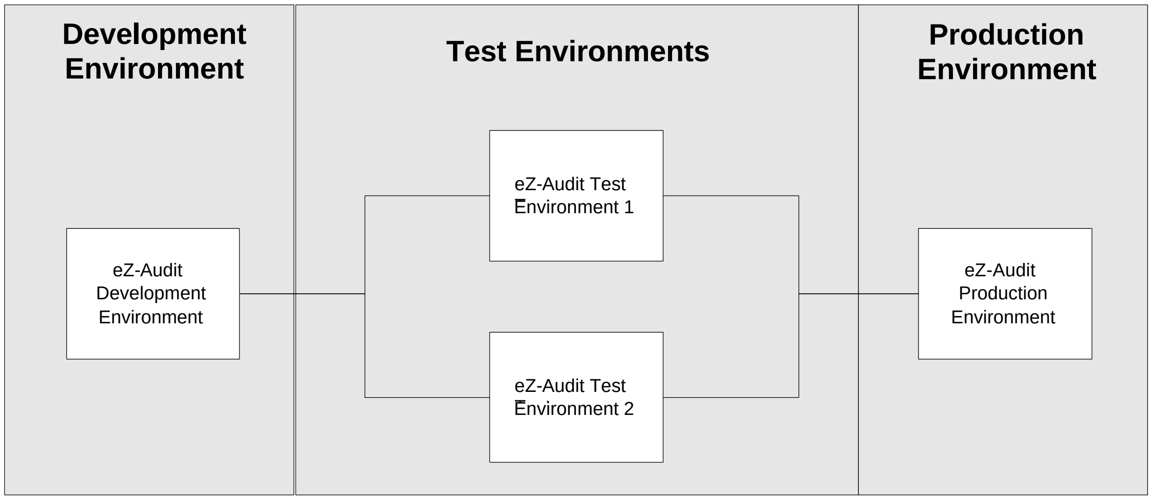

Development Environment
Test Environments
Production Environment
eZ-Audit Development Environment
eZ-Audit Test Environment 1
eZ-Audit Test Environment 2
eZ-Audit Production Environment

Figure 3, eZ-Audit Development, Test, and Production Environments

The following five environments will serve different purposes:

- *eZ-Audit Development Environment* – The Development environment will be created using the architecture for eZ-Audit and will be used for unit and assembly testing.
- *eZ-Audit System Test Environment* – The System Test environment will be created using the architecture for eZ-Audit and will be used for system and user acceptance testing. This environment will mirror the production environment.
- *eZ-Audit Production Environment* - The Production environment will be created using the architecture for eZ-Audit and will be used as the eZ-Audit production environment for the Deployment and Support project phases.

6 Testing Schedule

The test schedule section will cover the overall eZ-Audit test schedule, security, milestone information, and deliverable materials.

6.1 Overall Test Schedule

The eZ-Audit project workplan in Appendix A reflects the time duration of Unit Test, Assembly Test, System Test, Performance Test, and User Acceptance Test. The workplan includes tasks, percentage (%) complete, duration, start and end dates, and resource allocation.

6.2 Security

Once the initial System Test environment is created, the eZ-Audit Implementation Team will assist in the testing effort, but not have access to the System Test environment. Only members of the System Test team will have access to the environment. Migration of system components from the System Test team in the Configuration Management Tool to the System Test environment will be controlled and performed solely by the System Test Team. This will contribute to a valid System Test by ensuring that no one outside the System Test Team has access to the System Test environment. In addition, System Testers will have sole ownership of the System Test environment to ensure that new versions of code are not introduced during passes of test execution.

The testers conducting System and User Acceptance Tests will use a set of internal and external user IDs. This will allow the System Test team to verify the system security procedures and test connectivity. Additional security testing will be conducted to test access to the Development and System Test environments.

6.3 Milestone Chart

The following table depicts the activities and events to be conducted for Unit, Assembly, System, Performance, and User Acceptance Tests.

Test Phase	Planned Dates
Build and Unit Test	9/09 – 12/20
Test Plan Approach DUE	9/30
Test Script Development	10/01 – 11/08
Create Baseline Test Data	11/11 – 12/20
Test Environment Set-Up	12/02 – 12/20
Assembly Test	12/09 –12/20
Test Readiness Review (TRR)	12/20
System Test:	1/02 – 1/31
Test Pass 1	1/02 – 1/10
Code Fixes	1/02– 1/14
Test Pass 2	1/15 – 1/22
Code Fixes	1/15- 1/24
Test Pass 3	1/27 – 1/31
Code Fixes	1/27 - 2/05

Performance (Load) Test	2/03 – 2/21
User Acceptance Test (UAT):	2/03 – 2/21
UAT Prep	2/03 – 2/07
UAT Sessions	2/10 – 2/21
UAT Pass 1	2/11 – 2/13
UAT Pass 2	2/18 – 2/20
Final Code/Bug Fixes	2/24 – 3/14
Deployment/PRR	3/17 – 3/31
eZ-Audit Production Release	4/01/03

Table 10, Test Schedule

6.4 Deliverable Materials

The following Unit, Assembly, System, End-to-End, and User Acceptance Test deliverables will be delivered to the client for review as part of the eZ-Audit testing effort:

- Master Test Plan (Delivered prior to the testing effort)
- Test Results and Evaluation Report (Delivered after to the testing effort)

7 Open Action Items and Assumptions

This document provides an overall framework for system test execution, but is not a step-by-step instruction on how to execute System Test. The first execution of the System Test Plan is intended to validate the test plan, test data and system test environment. The following is a list of Assumptions and open Action Items that need to be resolved during or as a result of the first System Test execution.

7.1 Test Assumptions

7.1.1 Unit Test Assumptions

1. The Implementation team will perform unit testing without the use of test scripts. Instead, Developers will test individual code modules to determine whether the module meets the requirement(s).
2. The Implementation Team will document any deviations from the expected results.
3. All individual code modules must pass unit test successfully prior to moving to assembly test.

7.1.2 Assembly Test Assumptions

1. The Implementation team will perform assembly testing without the use of test scripts.
2. The Implementation Team will document any deviations from the expected results.
3. All unit-tested code modules must pass assembly test successfully prior to moving to system test.

7.1.3 System Test Assumptions

This document assumes the following for the duration of the testing effort. If any of these assumptions prove to be false, changes will result to the approach, the schedule, or both.

1. The Functional and Detailed Technical Design deliverables will accurately describe the workings of the system, and the software will adhere to the design.
2. The environment in which system testing will be performed will be fully available to the testing team for the duration of this effort, as defined in this document.
3. Any SIR not captured in the requirements or design documentation will be documented as an enhancement and staged for a future release, unless the Modernization Partner and FSA agree upon its inclusion.
4. The development, test, and production environments will be located at the VDC.
5. The development, test, and production environments will be available 24 hours per day x 7 days per week with the exception of planned maintenance efforts.
6. The development and test environments will mimic the production environment's hardware and software requirements.
7. The environments at the VDC will be accessible by all project resources (Implementation and Test).

8. Testers must have IDs with read and update access to the system. These IDs will allow testers to view contents of the database tables for verification of test results, and to manipulate the test data in the database in order to test the modules.
9. Test conditions will be defined and agreed upon within the Application Development phase.
10. A common tool will be used to capture System Investigation Requests. This tool will be developed using Rational ClearQuest.
11. The end-to-end test will only test limited transactions with the retrieval of the Schools file.
12. The end-to-end test will only test the data extracted from the Schools file that is relevant to eZ-Audit, based on specifications provided by the Modernization Partner team.
13. The end-to-end test will require that fixes to SIRs are re-tested using local copies of the original input files.
14. The end-to-end test will require that the Schools file being transferred resembles the data that will exist in production for eZ-Audit.

7.1.4 Performance Test Assumptions

1. System response time will be within acceptable tolerance (e.g., transactions should post in a matter of seconds, not minutes).
2. Load Runner will be used to mimic peak usage. eZ-Audit performance testing will test to ensure that the application can meet expected peak utilization plus a 20% contingency. This will be for both the web application and for batch processing.
3. The web application Performance Test will test both individual functional / screen performance and expected peak concurrent usage of the application. This will ensure that no there will be no significant reduction in system performance for peak usage.
4. The batch processes (PEPS School File Load, eZ-Audit-to-PEPS feed, FAC ACN file load, etc.) will be tested to ensure they can run in the batch window allotted to them.
5. The detailed Performance Test Plan will include the specific goals around each of these tests.

7.1.5 User Acceptance Test (UAT) Assumptions

1. UAT will be based on written, mutually agreed-upon scripts, which will be provided to FSA for review, feedback and sign-off. This effort will provide expected results, repeatable testing conditions, and defined exit criteria.
2. User groups will be available to execute testing in accordance with the approved schedule.
3. FSA will be responsible for organizing the internal and external participation for the user acceptance tests.

7.2 Open Action Items

The Action Items are broken up into three sections: System Test Plan, System Test Execution and System Test Environment.

7.2.1 System Test Plan

1. **Develop text for Static pages (e.g., Help content)**. The System Test Lead must work with the FSA Core Team to understand what static text needs to be created and then create the necessary test script(s) in order to test pages with static HTML.
2. **Work with Implementation Team to understand design changes and impacts to Test scripts**. It will be critical for System Test to communicate with the Implementation Team to understand if any design changes have been made and the corresponding effect the change has on the test scripts and test data. Likewise, testers must communicate any defects discovered during test execution to the System Test Lead in order to ensure the defect is being worked. The System Test Lead will be responsible for escalating any defects where appropriate.
3. **Conduct System Test Dry Run**. The System Test team may conduct a dry run of the System Test Cycles during Assembly Test to enhance the accuracy of the test scripts and the test data.

7.2.2 System Test Execution

1. **Create generic eZ-Audit logins to test functionality**. eZ-Audit login IDs and passwords will need to be set up for each user profile in order to login as each to test roles and responsibilities.
2. **Finalize baseline test data**. Additional baseline test data may need to be added depending upon the findings during the System Test Cycles. In the event that additional baseline data needs to be created, a separate System Test instance has been created. Coordinate with the Tech Arch Manager.
3. **Train testers on ClearQuest functionality**. All users must be set up with the ClearQuest application on their desktops in order to log system defects, or SIRs, discovered during test execution. Rational ClearQuest is the FSA standard used to log SIRs. A Ticket Master will need to be identified to track SIRs logged in ClearQuest and assign SIRs to the developers to ensure that all defects are being worked.

7.2.3 System Test Environment

1. **Confirm System Test Environment**. ITA will setup the Test Environment and test database.
2. **Configure System Test environment at the beginning of each Test Pass**. It will be necessary to migrate the latest version of code and data (e.g. JSP pages, ASP pages, HTML files, JPEG files, GIF images, XML, and style sheets) weekly in order to ensure the most recent versions of each are being tested.
3. **Coordinate data refreshes with the Tech Arch team**. The System Test Lead will be responsible for coordinating baseline data refreshes prior to the start of each Test Pass.

8 APPENDICES

8.1 Appendix A – eZ-Audit Workplan

See Attached MS Project File – eZ-Audit Workplan_09232002_v7.mpp

8.2 Appendix B – eZ-Audit System Test Schedule

See Attached MS Word File – eZ-Audit System Test Schedule_v1.doc

8.3 Appendix C – Sample Test Script

See Attached MS Excel File – Test Script Template v1.xls